2018

Apertium's Web Toolchain For Low-Resource Language Technology

S. Cherivirala

S. Chiplunkar

Jonathan North Washington
Swarthmore College, jwashin1@swarthmore.edu

See next page for additional authors

Apertium's Web Toolchain for Low-Resource Language Technology

Sushain Cherivirala sushain@skc.name
Independent Scholar, Apertium Community

Shardul Chiplunkar shardul.chiplunkar@gmail.com
Independent Scholar, Apertium Community

Jonathan North Washington jonathan.washington@swarthmore.edu
Linguistics Department, Swarthmore College, Swarthmore, PA 19081 USA

Kevin Brubeck Unhammer unhammer+apertium@mm.st
Trigram AS, Stavanger, Norway

Abstract

The Apertium web toolchain, consisting of a front end (Apertium HTML-Tools) and a back end (Apertium APy), is a free and open-source toolchain that supports a range of open-source technologies. The internationalised interface allows users to translate text, documents, and web pages, as well as morphologically analyse and generate text. Other features, including support for multi-step/pivot translation, dictionary-style lookup, spell-checking, and accepting user suggestions for translations, are nearing release.[1]

1 Introduction

Apertium APy[2] (API in Python) was begun in August 2013 as a drop-in replacement written in Python 3 for Apertium's previous query engine, ScaleMT[3], which was written in Java and was no longer maintained. Apertium HTML-Tools[4] was created later that year as a modern front end that interfaces with APy, replacing its less interactive predecessor. Both of these free and open-source (FOSS) applications constitute the Apertium web toolchain and have seen regular development and increased feature sets since their inception five years ago.

These tools were developed to make the FOSS language technology of Apertium (Forcada et al., 2011) available to a much wider audience than otherwise possible. Setting up the Apertium tools for use on a desktop operating system is a barrier to many who wish to use the tools, and their use on the command line can be cumbersome for tasks like translation, post-editing, and spell-checking.

[1]We appreciate support of this project by Google Code-In (2013–2017) and Google Summer of Code, and the time invested by GSoC students Kira Droganova (2016) and Monish Godhia (2017), as well as help from a number of contributors and translators.

[2]http://wiki.apertium.org/wiki/Apertium-apy

[3]http://wiki.apertium.org/wiki/ScaleMT

[4]http://wiki.apertium.org/wiki/Apertium-html-tools

Today, this infrastructure is deployed on the official Apertium website (apertium.org), the website for testing production and development Turkic-language tools (turkic.apertium.org), an "Apertium beta" site that makes available all of Apertium's language pairs regardless of development status (beta.apertium.org), and Giellatekno Apertium's translation site (jorgal.uit.no, maintained as a parallel branch). These sites allow anyone in the world with an Internet connection to make use of Apertium language technology.

APy is also used by Wikimedia Content Translation (Mistry et al., 2017), which facilitates the translation of content between Wikipedia articles in different languages, and the Sámi-language newspaper Ávvir[5], published in Norway, uses the spell- and grammar-checker back end for editing their publications. Similarly, Softcatalà, a non-profit association dedicated to fighting the marginalisation of the Catalan language, now employs APy as a translation service (Ivars-Ribes and Sánchez-Cartagena, 2011). Since the entire platform is FOSS, it is easily deployed on new systems and modified for specific uses.

This paper presents an overview of the web toolchain's architecture (§2), describes its core functionality (§3) and advanced features (§4), discusses on-going work (§5), summarises usage figures (§6), and concludes with thoughts on future work (§7).

2 Overview

The toolchain consists of a JavaScript/HTML/CSS front end called HTML-Tools and a Python 3.3+ back end called APy. The applications are type checked by Flow[6] and MyPy[7], respectively. The front end can function with any back end that supports the same API as APy; although almost all deployed versions of HTML-Tools are dependent on APy, it would be straightforward to have HTML-Tools use a custom back end developed for specific low-resource languages.

The back end can also be used as a general API for other purposes. For example, the IRC bot `begiak`[8] uses it to provide real-time translations and APy statistics, and the CAT tool OmegaT[9] has a plugin using APy.

HTML-Tools supports translation (of multiple text formats) and morphological functions in a fully internationalised environment. Currently the majority of the interface is localised in 25 languages. Responsive design makes the interface fluid on both mobile and desktop devices.

The machine translation (MT) endpoint of the API is, as with the ScaleMT system, similar to the Google Translate API, so MT consumers may easily switch between or support both APIs. Other endpoints support other functions, such as morphological analysis and generation, and provide localisation data to clients.

[5] https://avvir.no/
[6] https://flow.org/
[7] http://mypy-lang.org/
[8] http://wiki.apertium.org/wiki/Begiak
[9] https://omegat.org/

3 Core Features

The core feature of the Apertium web toolchain is the machine translation interface in HTML-Tools (Figure 1), which allows the user to choose a source and target language to translate their source text. Users may also use the language detection functionality powered by CLD2[10]. By allowing immediate access to three recently used languages, the interface facilitates switching between multiple frequently used languages.

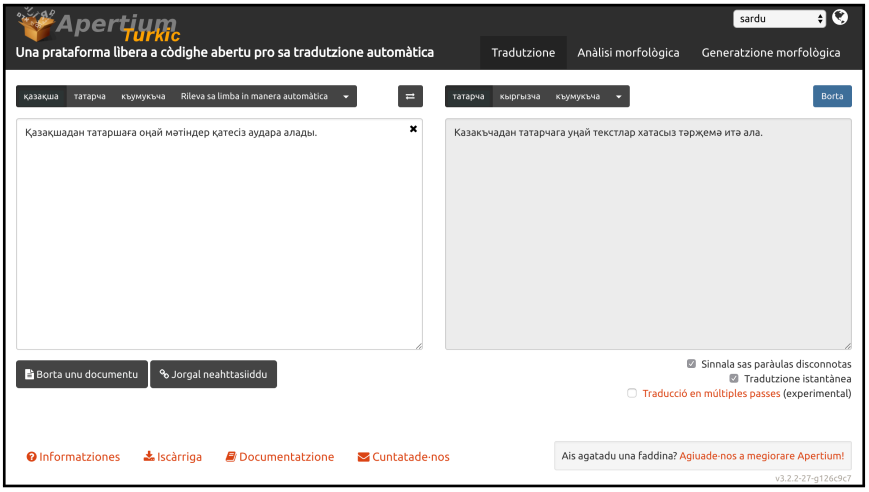


Figure 1: HTML-Tools' interface showing machine translation, and tabs for different modes. The screenshot also demonstrates localisation (Sardinian, with missing-string fallback to Northern Sámi) and a subtitle ("Turkic").

The toolchain offers a fully internationalised experience with the HTML-Tools' interface language defaulted to match the user's browser locale and manually controllable by a language selector. Both right-to-left and left-to-right scripts are supported. The interface's string localisations are located in JSON files, one for each language, that each contain some metadata and a simple key-value storage schema with support for basic templating. In addition to the interface's strings being localised, glossonym localisation is powered by APy where language names are fetched from a SQLite database. The database is populated from text files containing data from SIL International[11] and the Unicode Common Locale Data Repository[12] as well as manual curation[13]. Autoglossonyms (and following that, ISO 639-3 three-letter codes) are used as fallbacks when a language name is not localised in the interface's current language.

In contrast to many modern web applications, HTML-Tools eschews complex build dependencies and tools such as Webpack, requiring only GNU Make, `curl`,

[10] https://github.com/CLD2Owners/cld2
[11] https://www.sil.org/
[12] http://cldr.unicode.org/
[13] These scripts are bundled with APy.

and Python 3 in a standard POSIX environment to successfully build its static resources which can then be served by any web server. Performance optimisations such as resource compression are entirely optional and offline building and usage are supported.

APy is modeled after the ScaleMT infrastructure (Sánchez-Cartagena and Pérez-Ortiz, 2010). Every translation language pair (e.g., Catalan to Spanish) or monolingual analysis/generation pipeline corresponds to an Apertium `mode`, which is a Unix pipeline defined by the Apertium language data developers. Since pairs involve multiple executables running in serial accessing large binaries, it would be prohibitively slow to bootstrap a pair on each request, so pipelines are kept open between requests and flush data upon seeing a `NUL` character (which is added at the end of each request).[14]

The server is typically run in a single Python process using the Tornado library[15], which uses green threads to allow large numbers of asynchronous/non-blocking requests. Large requests are split into manageable sizes so they do not block the server even if other requests to the same `mode` come in. Like ScaleMT, APy allows opening several copies of the same `mode` in case of high traffic, and shutting down unused ones.

The process handling is general enough that it can make any Unix pipeline into a scalable, robust, non-blocking web service, as long as the pipeline can be made to flush output on seeing a certain input. The spell- and grammar-checking pipeline used by Ávvir (which does not use APy's built-in spelling backend) is one example of taking a "new" pipeline and using APy to turn it into a web service.

4 Advanced Features

The toolchain has first-class support for language variants, a feature particularly relevant to some low-resource languages. Within APy, all endpoints accept language codes with variants, e.g. `oci_aran` represents Aranese, a variety of Occitan. HTML-Tools provides special rendering to variants, as shown in Figure 2 where variants are always nested within their 'parent' languages to aid discoverability.

In HTML-Tools, all user inputs and selections are persisted in their browser's local storage unless disabled, maintaining the interface's consistency between page reloads and prevent accidental data loss. By synchronising the browser's displayed URL with user inputs, users can share their URL or bookmark it to reach the same translation.

In addition to text translation, the toolchain supports web page and document translation. In HTML-Tools, URLs are automatically detected in the source text input and APy handles the fetching and translation of the URL, the result of which is displayed within a `iframe` in HTML-Tools. Any links in the web page are instrumented to also trigger translation. APY's document translation endpoint supports standard text formats including LibreOffice and Microsoft Office.

[14] A technique pioneered by Wynand Winterbach.

[15] http://www.tornadoweb.org/en/stable/

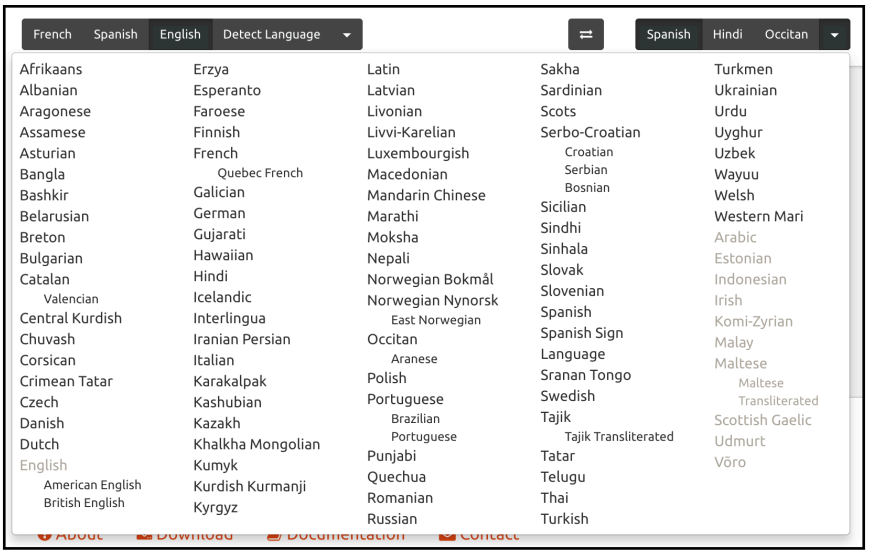


Figure 2: Possible target languages for a translation from English when multi-step translation is enabled (see §5.1) on beta.apertium.org.

Aside from the translation mode, HTML-Tools provides other modes visualised as tabs in the interface. Two of these are morphological analysis and generation. The output of morphological analysis has pretty-printing support, and morphological generation of surface forms accepts analyses in Apertium stream format[16]. Another tab is a sandbox mode that facilitates querying APy with arbitrary content, a particularly useful tool for developers. Navigation between modes is synchronised with the browser's URL to ensure consistency for reloads and URL sharing.

HTML-Tools has built-in integration with Matomo (formerly Piwik)[17], a free and open-source web analytics platform, to enable collection of statistics such as which language pairs are most often used (see §6). In a similar vein, APy supports not only the logging of usage statistics, but also the collection of words in translation requests that are unknown to Apertium's translation engine. These words have the potential to serve as seed data for future initiatives aimed at improving language pair performance.

To aid in development, the toolchain is currently configured with linters for JavaScript, HTML, CSS, and Python. These linters and a basic test suite for APy are run via continuous integration platforms CircleCI[18] (HTML-Tools) and Travis CI[19] (APy). A Docker[20] configuration is provided to enable starting the entire toolchain with a single command.

[16] http://wiki.apertium.org/wiki/Apertium_stream_format
[17] https://matomo.org/
[18] https://circleci.com/
[19] https://travis-ci.org/
[20] https://www.docker.com/

5 Ongoing Work

Several features are in progress and have little work remaining, primarily consisting of merging changes from various contributors into the toolchain and ensuring that the features do not interfere with each other.

5.1 Multi-step Translation

Multi-step translation, i.e. translation with one or more intermediate languages, is supported by APy. An APy request can specify the precise path for translation or can specify just the ultimate source and target languages and allow APy to select an appropriate path.

Currently, multi-step translation involves piping the generated text of one pair into the analyser of another, possibly introducing surface form ambiguity; future work could improve this by bypassing the intermediate generators and analysers and directly passing the morphological analysis between language pairs. Also, when no path is specified, APy chooses a translation path solely by minimising the number of intermediate languages; future work could improve this by introducing some numerical measure of the quality of a pair and hence enabling APy to choose the qualitatively 'best' path.

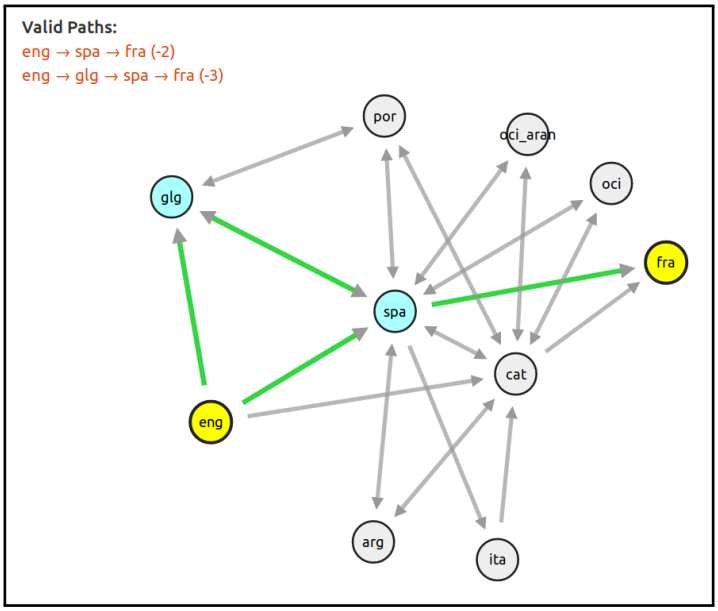


Figure 3: Graphical interface to choose a multi-step translation path from English to French using Apertium's released pairs. Blue nodes are intermediate languages that the user has selected, and green arrows form valid translation paths through those nodes.

A basic interface for multi-step translation has also already been implemented

in HTML-Tools[21]. When enabled, multi-step translation allows the user to select any target reachable via a multi-step path from the selected source. However, this approach does not provide any information about or control over the chosen path. To remedy this, an graphical multi-step interface has been developed in HTML-Tools (unreleased). Figure 3 shows a typical interface presented to the user upon selecting English as the source language and French as the ultimate target language. Further, the nodes in the graph are draggable, and information about the selected path is also represented elsewhere in the translation interface.

5.2 Dictionary Lookup

Figure 4 illustrates the dictionary lookup feature of the translation interface. When a translation is requested for a single word, HTML-Tools uses APy's dictionary endpoint to fetch all possible translated lemmas along with their part-of-speech.

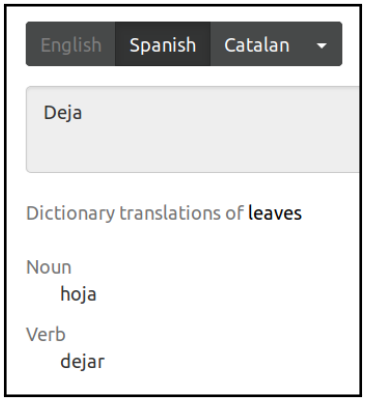


Figure 4: Dictionary lookup interface showing possible Spanish translations of the English word 'leaves'.

This functionality has been implemented on feature branches in APy and HTML-Tools, but requires further work prior to release. Specifically, the dictionary mode could provide reverse translations for additional context, grammatical information such as the gender of nouns or the conjugation paradigms of verbs, and information about multi-word lexical units which are currently not handled by HTML-Tools although APy supports dictionary lookup for any lexical unit.

5.3 Spell-Checking

The spell-checking mode of HTML-Tools allows users to spell-check input text in languages that support the feature. The interface is separate from the interfaces for translation, analysis, etc. In APy, spell-checking relies on a speller mode being enabled in a language module; these modes often use libvoikko[22] or hfst-ospell[23].

[21] Enabled, for example, on `turkic.apertium.org`.
[22] https://github.com/voikko/corevoikko
[23] https://github.com/hfst/hfst-ospell

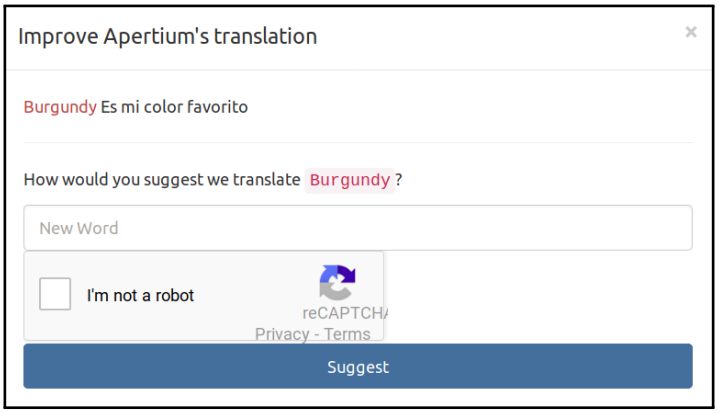
Generating a mode from an existing language module is fairly simple, requiring only installation of the libraries and tools and small additions to the Makefile.

Implementations of spell-checking on the front and back end are still undergoing testing for robustness and usability, with plans to release them soon.

5.4 Suggestions

The suggestions interface allows users to suggest translations of unknown words, as shown in Figure 5. In APy, support for suggestions is still in development, while the HTML-Tools interface is developed but needs some refinement before release.

(a) An unknown word is highlighted in the output of a Spanish-English translation, and the interface provides an opportunity to offer a suggestion.



(b) An interface is provided for offering a suggestion. Context is included in the submitted suggestion.

Figure 5: When suggestions are enabled and an unknown word is encountered, users can suggest translations of the word that are sent to an APy endpoint.

Another proposed improvement is providing users the ability to rate translations as 'thumb up/down' or on a numerical scale. This requires less effort for the users and is thus likely to produce more feedback for Apertium, and the numerical ratings could be used as a measure of quality for pairs produced by human evaluation. Such a qualitative measure would be useful for many Apertium applications including multi-step translation path selection (§5.1).

6 Usage Statistics

As discussed in section 4, HTML-Tools supports web analytics via Matomo. In Table 1 we present some statistics from the apertium.org site to testify to the robustness of the toolchain and show trends in end-user behavior and demographics.

Since April 2014, the site has served 2.1 million visits from 183 distinct countries around the world. During this period, the associated APy instance received 19.8 million translation requests. Only ~0.5% of the requests were for document or web page translation, the only other functions exposed on apertium.org.

Table 1 lists the language pairs that have received over one hundred thousand requests. We note that instant translation is enabled by default and typing into the source text input continuously will intermittently trigger translation requests.

Language Pair	Requests (thousands)		Characters (millions)	
nob-nno	12,286	62.3%	7,225	16.8%
spa-cat	2,005	10.2%	7,083	16.5%
nno-nob	726	3.7%	758	1.8%
por-spa	693	3.5%	519	1.2%
spa-cat_valencia	672	3.4%	2,344	5.5%
cat-spa	652	3.3%	9,197	21.4%
eng-spa	544	2.8%	5,376	12.5%
spa-por	318	1.6%	679	1.6%
spa-eng	286	1.5%	1,087	2.5%
eng-cat	151	0.8%	990	2.3%
nob-swe	126	0.6%	62	0.1%

Table 1: Translation requests served by apertium.org grouped by language pairs (using ISO 639-3 codes). Percentages indicate portion of all requests.

7 Conclusion

Perhaps the most important reason why Apertium's web toolchain is well-suited for low-resource languages is that the toolchain enables easy public access to language technology with very low costs and maintenance requirements, allowing developers to spend more funding and time on developing the technology itself. All the software components required to run an online language service are free and open source. Further, their disk, memory, and processing requirements are low enough to work on any personal computer. Once downloaded, even an Internet connection is not required to use these tools.

As mentioned in §2, HTML-Tools provides a free, open source, and customisable interface for custom low-resource language services. The web interface also allows for sub-sites showcasing tools for low-resource languages. An example of such a sub-site is turkic.apertium.org for the Turkic languages in Apertium.

Lastly, features described in §5 have the potential to be greatly beneficial for low-resource languages. Multi-step translation, if used with existing high-

quality translation pairs, can produce moderate-quality pairs using intermediate languages with no extra effort, extending the utility and range of possible translations among low-resource languages. The suggestions interface can make it very easy for users of low-resource languages and technology to help their developers improve these tools.

As a free and open-source project, Apertium is driven by its community. We welcome all suggestions, feedback, and pull requests! The HTML-Tools GitHub repository[24] and the APy GitHub repository[25] have their own issue/pull request trackers, while comments about language data or questions about installation are welcome on Apertium's mailing list[26] and Freenode IRC channel, `#apertium` [27].

Beyond technical contributions, we also appreciate help improving HTML-Tools' localisation by revising or extending current ones, or adding new ones.

As current usage by Apertium and other organisations demonstrates, the Apertium web toolchain features a platform that enables end users to quickly benefit from the efforts of mature language technology. A host of improvements in the pipeline from spell-checking to dictionary lookup and a steady stream of contributors signal a promising future.

References

Forcada, M. L. et al. (2011). "Apertium: a free/open-source platform for rule-based machine translation". In: *Machine Translation* 25 (2), pp. 127–144.

Ivars-Ribes, X. and V. M. Sánchez-Cartagena (2011). "A Widely Used Machine Translation Service and its Migration to a Free/Open-Source Solution : the Case of Softcatalà". In: *Proceedings of the Second International Workshop on Free/Open-Source Rule-Based Machine Translation*. URL: http://hdl.handle.net/10609/5648.

Mistry, K. et al. (2017). *Content translation/Machine Translation/Apertium/Service*. URL: https://www.mediawiki.org/wiki/Content_translation/Machine_Translation/Apertium/Service (visited on 2018-02-10).

Sánchez-Cartagena, V. M. and J. A. Pérez-Ortiz (2010). "ScaleMT: a free/open-source framework for building scalable machine translation web services". In: *The Prague Bulletin of Mathematical Linguistics* (93), pp. 97–106.

[24] https://github.com/goavki/apertium-html-tools
[25] https://github.com/goavki/apertium-apy
[26] https://lists.sourceforge.net/lists/listinfo/apertium-stuff
[27] http://wiki.apertium.org/wiki/IRC